TITLE OF THE INVENTION

DIGITAL COMPOSITION OF A MOSAIC IMAGE

CROSS REFERENCE TO RELATED APPLICATIONS

A claim of priority is made to U.S. Provisional Patent Application Serial No. 60/035,733, filed January 2, 1997, entitled: DIGITAL COMPOSITION OF A MOSAIC IMAGE.

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH OR DEVELOPMENT

Not applicable

BACKGROUND OF THE INVENTION

The present invention is generally related to computerized manipulation of images, and more particularly to generation of an image from a plurality of sub-images.

Analysis and manipulation of images using computers is well known. For example, computers have been used to analyze images of coins travelling along a conveyor belt to distinguish different types of coins and compute the total value of the coins. Similarly, computers have been used to analyze images of integrated circuits and printed circuit boards in order to detect defects during manufacturing. Manipulation of photographic still images and full motion video images to produce special effects is also well known. However, these known techniques do not produce artistically pleasing mosaic images.

WEINGARTEN, SCHURGIN, GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

BRIEF SUMMARY OF THE INVENTION

In accordance with the present invention, a mosaic image that approximates a target image is produced from a database of source images by analyzing tile portions of the target image, comparing each respective analyzed tile portion of the target image with the source images from the database to provide a best-fit match in accordance with predetermined criteria, and generating a mosaic image comprising the respective best-fit match source images positioned at respective tile portions of the mosaic image which correspond to the respective analyzed tile portions of the target image. In one embodiment the criteria for the best-fit match includes computing a version of Red, Green and Blue ("RGB") Root-Mean Square ("RMS") error. Other matching systems could be employed as long as the goal of finding the source image that is most visually similar to the region of the target image under consideration is met.

Increased resolution is realized in the mosaic image through sub-region analysis. In particular, each tile portion in the target image is divided into sub-regions which are independently compared with corresponding sub-regions of each source image using, in this example, RGB RMS error analysis. The computed RGB RMS error for each sub-region is summed to provide a sum RGB RMS error for the entire source image. The unallocated image having the lowest sum RGB RMS error is then allocated for use in the corresponding tile portion in the mosaic image. The use of sub-regions even benefits regions without detail and results in more uniform distribution of color by selecting lower contrast images for these areas of little high-frequency detail. Another embodiment employs a second pass to prevent a source image from being placed in a given location in the mosaic if it would have a lower error in another location.

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

insert E1

BRIEF DESCRIPTION OF THE DRAWING

The invention will be more fully understood in view of the following Detailed Description of the Invention, in conjunction with the Drawing, of which:

Fig. 1 is a block diagram of a mosaic image generating system;

Fig. 2 is a block diagram of a database of source images;

Fig. 3 is a flow diagram that illustrates a method of mosaic image generation;

Fig. 4 is a diagram that illustrates tiles and sub-regions;

Fig. 5 illustrates the effect of sub-region analysis on source image selection; and

Fig. 6 illustrates the effect of sub-region analysis on final mosaic image resolution.

DETAILED DESCRIPTION OF THE INVENTION

Fig. 1 illustrates apparatus for generating a mosaic image 10 from captured source images 12 to approximate a target image 14. In the disclosed embodiment a VHS video tape player 16 is employed to facilitate capture of source images from video tapes. The video tape player may be employed to single-step through a video tape to capture still images for use as source images. Alternatively, source images can be captured in real-time during playback of a video tape. A computer controllable laserdisc player 18 also can be employed to facilitate capture of source images. Laserdiscs are preferable to video tapes when the desired subject matter is available from both sources because of the higher quality and easy random access to still images available from laserdisc. In the disclosed embodiment a computer workstation 20 with a video input is employed to capture the source images 12 from the video tape player 16 and laserdisc player 18. The computer workstation also accepts the target image 14 as input, and is employed to

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

4

generate the mosaic image 10 from the target image and source images by executing mosaic software. The mosaic image 10 generated by the mosaic software comprises an array of tiles 22, where each tile 22 is a source image 12, and the overall appearance of the mosaic image 10 approximates the appearance of the target image 14. An editing computer 24 such as a Macintosh (TM), PC or UNIX (TM) based system equipped with image editing software such as Adobe Photoshop (TM) can be employed for editing the mosaic image 22, to produce an edited mosaic image 26. A printer output device 28 may be employed to print the edited mosaic image 26.

Captured source images 12 can be analyzed and stored in a database 30 that is maintained in the workstation 20. An add_images_to_database program is employed to analyze raw captured source images 12 and create new source images therefrom. More particularly, the add_images_to_database program accepts a list of filesystem directories, an image size, and an output path as input, and operates in response to open each designated directory and search for source images from which to crop and resize to the specified dimensions. The square is subsequently moved to the location specified by the output path. In one embodiment, if the source image is in landscape format, a square image is cropped from the center of the source image. If the source image is in portrait format, a square is cropped from between the center and the top of the source image. As a consequence, the square image is more likely to include the emphasized feature of the source image, such as a person's face, without clipping the edges thereof. The images are then stored in the database 30. The database 30 is a file system which holds the formatted images in directories that are categorized by subject matter and size.

Fig. 2 illustrates organization of source images 12 within the database 30 (Fig. 1). Source images 12 are categorized and placed under root nodes such as an animals root node 32, a people root node 34 or a places root node 36. To generate a mosaic image from source images of animals, the

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

animals root node 32 is selected for the mosaic software. Directly under the animals root node are subdirectories containing identical image files at different levels of resolution. An originals subdirectory 38 contains uncropped versions of each source image file at full size 40. The originals subdirectory 38 is maintained because source images may be recropped during mosaic creation if the results from the add_images_to_database program are unacceptable. Directories labeled 256 x 256 (pixels) 42 and 64 x 64 (pixels) 44 contain large versions of the formatted source images which are used primarily for outputting a final bitmap. In this example, a 32 x 32 (pixels) 46 directory contains source images which are used for viewing the mosaic image on the screen during the construction process. The 16 x 16 (pixels) 48, 8 x 8 (pixels) 50, and 1 x 1 (pixels) 52 subdirectories contain source images which are preloaded when the mosaic software is initialized. The source images in the 16 x 16, 8 x 8, and 1 x 1 subdirectories are employed for matching source images to target image during mosaic image generation. Directories of source images at other levels of resolution may also be maintained.

Fig. 3 illustrates a method for generating the mosaic image. Referring now to Figs. 2, 3 and 4, the target image is selected and loaded as indicated in step 60. A root node of source images in the database is then selected and loaded as indicated in step 62. More particularly, a database path is specified and a mosaic program is executed. The mosaic program reads source images from the section of the database indicated by the specified database path, analyzes the target image and selects a source image for use in each tile of the mosaic image. More particularly, source images having resolution corresponding to the selected number of sub-regions ("sub-region resolution") for the mosaic image are loaded into a linked list of structures:

```
struct an_image {
    char *path;          /pathname of file in database/
```

1,060

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

```
        char used;          /whether image has been used/
        unsigned short *r;  /RGB image data for RMS matching/
        unsigned short *g;
        unsigned short *b;
        struct an_image *next;    /pointer to next structure/
        struct an_image *previous; /pointer to prev structure/
} an_image;
```

For example, if each tile in the mosaic image is to contain 8 X-axis sub-regions by 8 Y-axis sub-regions, then 8 X 8 (pixels) images are loaded from the database. The size of the target image in pixels along each axis is equal to the number of output tiles multiplied by the number of desired sub-regions to be considered during the matching process, i.e., one pixel per sub-region along each respective axis. The respective numbers of tiles which will be employed for the X and Y axes of both the mosaic image and target image is then specified as indicated in step 64.

The mosaic program executes a matching process once the source and target images have been loaded. When the matching process begins, the target image is divided into "x" by "y" tiles 22, where (x, y) is:

(target_image_width / width_subsamples, target_image_height / height_subsamples)

A new tile is loaded as indicated in step 68. A new sub-region 66 is then loaded as indicated in step 70. Loading begins with the upper left sub-region 66 of the tile 22, and moves from left to right through each row, and from top to bottom by row. The source image pixel that corresponds to the loaded sub-region is then loaded as indicated in step 72.

The matching process analyzes tiles 22 individually on a serial basis. For each tile 22 in the disclosed embodiment, a variation of the average Root-Mean Square ("RMS") error of the Red, Green, and Blue ("RGB") channels of each sub-region 66 is compared to each respective

WEINGARTEN, SCHURGIN, GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

corresponding source image pixel, for each source image in the database that is of proper resolution and is not designated as "used." A RMS error between the loaded pixel and loaded sub-region is computed for RGB channels and kept as a running sum for the tile as indicated in step 74. If unanalyzed sub-regions exist in the tile as indicated in step 76, flow returns to step 70. If all sub-regions have been analyzed, as determined in step 76, then the running sum RGB RMS error is compared to the lowest such error yet computed for a source image and the tile as indicated in step 78. If the error sum is lower than any previously recorded error sum for the tile, the error sum value and an index to the source image are recorded as indicated in step 80.

When all of the source images have been analyzed for similarity to the tile, the source image with the least computed RGB RMS error is assigned to a tile in the mosaic image corresponding to the tile in the target image, i.e., in the same location in the image. More particularly, if other source images in the database have not been compared with the tile as determined in step 82, a new source image is loaded as indicated in step 84 and flow returns to step 70. If all source images have been compared with the tile as determined in step 82, the source image with the lowest sum error is allocated to the tile and marked as "used" as indicated in step 86. The assigned source image is marked as "used" so that source images do not appear more than once in the mosaic image.

The matching process is repeated for each and every tile in the target image. Upon completion, a list of source images is written to a text file which is used by a final rendering program to construct a bitmap from the full resolution versions of the source images. More particularly, if all tiles have been examined as determined in step 88, a list of the lowest sum error source images for each tile is written to a text file as indicated in step 90, and the mosaic program reads the list and assembles a bitmap as indicated in step 92. If unexamined tiles still exist as

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

determined in step 88, flow returns to step 68.

A variation of the matching process, including computation of RMS error, is implemented as follows:

```
/* The goal of this routine is to find which source photographs are the most */
/* visually similar to a given region (grid-space) of the target image. */

int find_matches(int x, int y)
{
     register i, rt, gt, bt;
     int low, result, ii, the tile;
     char imagename[256], best_path[256];
     unsigned short rmas[XMAX*YMAX], gmas[XMAX*YMAX], bmas[XMAX*YMAX];

     the_tile = x+(y*sizex);

     /* For this given grid-location of the target image, clear the list of errors. */
     /* This list will later contain the computed errors and will be sorted from best */
/* to worst */

     for(i = 0; i < pixels; i++)  {
          tiles[the_tile].list[i].score=99999999;
          tiles[the_tile].list[i].rank = 0;
     }

     strcpy(imagename, filename);  /* Get the name of the target image */

imagename[strlen(imagename)-3] = 's'; /* Make sure that it has the proper filename extension */
imagename[strlen(imagename)-2] = 'g';
     imagename[strlen(imagename)-1] = 'i';

get_grid_space(rmas, gmas, bmas, x, y); /* Get the image data for the desired region of the */
          /* target image and put it into three arrays. */

image = head_image;     */ Reset the linked-list of source images to the beginning */

while(image->next != NULL) {   /* For every source image we are considering */

     result = 0;

/* This is a variation of RGB RMS error. The final square-root has been eliminated to */
/* speed up the process.  We can do this because we only care about relative error. */
```

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

```
/* HSV RMS error or other matching systems could be used
here, as long as the goal of */
/* finding source images that are visually similar to the
portion of the target image */
/* under consideration is met. */

for(i = 0; i > size; i++) {
     rt   =   (int)((unsigned   char)rmas[i]   -   (unsigned
     char)image->r[i]);
     gt  =  (int)((unsigned char)gmas[i]  -  (unsigned char)
     image->g[i];
     bt   =   (int)((unsigned   char)bmas[i]   -   (unsigned
     char)image->b[i];
     result += (rt*rt+gt*gt+bt*bt);
}
i = 0;

     /* The following code takes the error computed for the
last source image and inserts */
     /* it into a sorted list of all of the source images.
The list is shifted towards the */
     /* end to make room for this insertion */

     if (result < tiles[the_tile].list[pixels-1].score) {
          while((result   >   tiles[the_tile].list[i].score)
&&(i++ < pixels));

     for(ii = pixels-1; ii> i; ii--) {
          tiles[the___tile].list[ii}.score    =    tiles[the
          tile].list[ii-1].score;
          tiles[the____tile].list[ii].rank    =    tiles[the
          tile].list[ii-1].rank;
          tiles[the___tile].list[ii].pointer   =   tiles[the
          tile].list]ii-1].pointer;
}
          tiles[the_tile].list[i].score = result;
          tiles[the_tile].list[i].rank = i;
          tiles[the_tile].list[i].pointer = image;
}
     /* Now let's move to the next source image and repeat
     until we run out */

     image = image->next;

     } /* while */

/* Since the list is sorted from next to worse, we can see
the best tile by looking at */
/* the first list entry. */

low = tiles[the_tile].list[0].score;
tiles[the_tile].score = tiles[the_tile].list[0].score;
tiles[the_tile].rank = tiles[the_tile].list[0].rank;

strcpy(best_path, tiles[the_tile].list[0].pointer->path);
```

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

```
/* Do not let this image get replaced later because it was
specified as required for the mosaic.*/
tiles[the_tile].required = tiles[the_tile].list[0].pointer-
>required;

strcpy(tiles[the_tile].path, best_path);
sprintf(imagename, "%s/%s", disp_version, best_path);

/* We now have a sorted list of source images from most-
visually-similar to least-visually-similar */
/* for this grid location of the target image.*/

return low;

} /* find_matches () */
```

A second routine is used in one embodiment of the invention to take the sorted list from the previous routine and not only ensure that each source image is only used once but also to see that a given source image will not be selected for one region if it is an even lower match in another.

```
     /* In the first phase of the program (find_matches ()),
e created a sorted list of source images */
     /* for each grid-space of the target image.  Since we
do not want to repeat source images within */
     /* the mosaic, each grid-space cannot have its first
choice source image (a source image may have */
     /* the lowest match for more than one grid location).
The purpose of this routine is to decide which */
     /* of the grid locations actually gets to use the source
image.  For example, it will not be placed */
     /* in one grid location if it an even better match to
another */

int optimize ()
{

     int i, x, deepest = 0, change, a, step, which;

     /* For each of the grid-locations in the target image
(number of tiles in the final mosaic) */
     /* This an N^2 algorithm, so we must loop twice to
ensure that we consider all images for */
     /* all grid-locations. */
     for(a = 0; a < pixels; a++) {
          change = 0;

          /* For each of the grid-locations in the target
image (number of tiles in the final mosaic) */
for(x = 0; x < pixels; x++) {
     which = 0;
```


WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

```
        do {
             step = 0;
             for(i = 0; i < pixels; i++) {

                  /* If tile is wanted more somewhere else, give
        it to them. */
                  /* We do this by going through all the top
        choices for the other grid locations. */
                  /* If we see the same source image listed as
        the first choice at another grid */
                  /* location, we check to see if it is a better
        match at the other location. */
                  /* If it is, we move through our sorted list
        to the next best match for our current */
                  /* grid-location and do this until we find a
        source image that is not a better match */
                  /* anywhere else.  When we find this, we can
        keep it.  The variable "step" stays as 0 and */
                  /* we exit the do-while loop */
                  if    ((tiles[i].    rank    <=    which)    &&
        (!strcmp(tiles[x].list[which].pointer->patch,
        tiles[i].path))) {
                       /* If rank is same, check scores. */
                       if    ((tiles[i].rank    ==    which)    &&
        (tiles[i].score > tiles[x].list[which].score)) continue
                       if (i == x) continue;
                       which++;
                       step = 1,
                       i - pixels; /* Skip to while. */
                  }
             }
        } while (step);

        if (which > deepest) deepest = which;

        /* Now that we found the most visually-similar source
        image that is *not* a better match in another */
        /* grid location, we se the name of the image as
        associated with this grid-location of the target */
        /* image. */

        if (strcmp(tiles[x].path, tiles [x].list[which].pointer-
        >path)) {
        change++;
        strcpy(tiles[x].path,    tiles[x].list[which].pointer-
        >path);
        tiles[x].required    =    tiles[x].list[which].pointer-
        >required;
        tiles[x].score = tiles[x].list[which].score;
        tiles[x].rank = which;
        }

        } /* for */

        /* If we go through all of the grid-locations and we do not
        need to replace any */
```

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

```
/* tiles as being a better match in another location, we can
exit the routine now. */
if (!change) break;

fprintf(stderr, "\n%d/%d, %d changes (deepest is %d)\n", a,
pixels-1, change, deepest);

/* We need to loop back with this for loop as many times as
there are grid-space in the final mosaic. */ } /* for */

} /* optimize() */
```

A rendering program can be employed to produce the mosaic image following the matching process. The rendering program reads the list of the selected tiles, locates the full sized version of each respective corresponding source image in the database, and binds the located source images together to create a bitmap. The tiles in the mosaic image may be separated by a line to discretize them when viewed from close proximity. From a distance, the gridlines should be thin enough to disappear completely to the human eye, so as not to interfere with the seamlessness of the mosaic. The bitmap is then saved in a standard format to be displayed on a monitor or output in printed form.

The digital mosaic image can be printed in different ways, depending on quality, price and size constraints. Film recording and photographic printing may be employed. An image can be written to photographic film using a film recorder. Once the image is on chrome or negative, it can be printed on normal photographic paper. This option is best for a moderate number of small copies as writing the image onto the film is a one time cost. Direct digital printing potentially produces the highest quality, but each print is expensive. Digital printers employ either continuous-toning or half-toning. Continuous-tone printers deposit an exact color for each pixel in the image. Half toning printers deposit only drops of solid color, forming shades of color by using dots of different sizes or different spacing. Hence, the print will look less photographic. Process color printing is the technique used to reproduce images in magazines and books, and is a good method for producing many

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

(e.g., hundreds of thousands) near-photographic copies.

The effects of sub-region based analysis on source image selection are illustrated in Fig. 5. A target image 100 was employed to produce first, second and third mosaic images 102, 104, 106, respectively. The target image 100 includes 4 X 4 tiles. An intermediate "sensed" image representing the average of all pixels in the smallest analyzed portion (tiles in image 108, and sub-regions in images 110 and 112). In the first analysis, resulting in images 108 and 102, sub-regions are not employed. In the second analysis, resulting in images 110 and 104, 4 X 4 sub-regions per tile are employed. Because some light and dark regions can be sensed within each tile in the second analysis, those sensed regions are taken into consideration when searching the database during the selection process.

In the third analysis, resulting in images 112 and 106, 16 X 16 sub-regions are employed. With 16 X 16 sub-regions, the intermediate image 112 is substantially closer to the target image 100. Further, image 106 shows that when this amount of detail is considered during the selection process, more appropriate matches are selected. For example, the woman in the first row is the same shape as the vertical black bar in the same region of the target image. Further, the lizard in another tile matches the diagonal that it was compared to. This high-degree of shape matching has a powerful effect on the image-forming ability of the final mosaic image as information about the contours and shading in a target image may transcend the boundaries of each mosaic tile.

In addition to providing improved source image selection, the use of sub-regions results in more uniform distribution of color by selecting lower contrast images for regions of little high-frequency detail. This can be seen in the lower eight tiles of image 106 which are more uniform than those selected for image 104.

Fig. 6 illustrates the effects of number of sub-regions on mosaic image resolution. First and second mosaic images

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313

144, 116 were generated from a target image 118. The first mosaic image 114 was generated with 2 x 2 sub-regions within each tile considered during the source image selection process. The second mosaic image 116 was generated with 16 x 16 sub-regions within each tile considered during the source image selection process. The same collection of source images was employed to produce both the first and second mosaic images. Because of the sub-region analysis, different source images were selected to represent some corresponding tiles in the first and second mosaic images. Further, the second mosaic image 116 bears a stronger resemblance to the target image 118 than the first mosaic image 114. Hence, improved source image selection provided through analysis of more sub-regions generates improved resolution in the resultant mosaic image.

In an alternative embodiment, semantic content is specified for portions of the mosaic image. More particularly, image sub-categories are specified for use with specified tiles of the target image. Hence, the resultant mosaic image includes tiles or regions of tiles with predetermined categories of images.

In another alternative embodiment images can be selected for assured selection and inclusion in the mosaic image. More particularly, the selected images are placed in the location of greatest visual similarity relative to the target image even if another (unassured) image is determined to have greater visual similarity.

Having described the preferred embodiments of the invention, other embodiments which incorporate concepts of the invention will now become apparent to one of skill in the art. Therefore, the invention should not be viewed as limited to the disclosed embodiments but rather should be viewed as limited only by the spirit and scope of the appended claims.

WEINGARTEN, SCHURGIN,
GAGNEBIN & HAYES LLP
TEL (617) 542-2290
FAX (617) 451-0313